On Complexity, Energy- and Implementation-Efficiency of Channel Decoders

Frank Kienle, *Member, IEEE,* Norbert Wehn, *Senior Member, IEEE*, and Heinrich Meyr, *Fellow, IEEE*

arXiv:1003.3792v1 [cs.IT] 19 Mar 2010

Abstract

Future wireless communication systems require efficient and flexible baseband receivers. Meaningful efficiency metrics are key for design space exploration to quantify the algorithmic and the implementation complexity of a receiver. Most of the current established efficiency metrics are based on counting operations, thus neglecting important issues like data and storage complexity.

In this paper we introduce suitable energy and area efficiency metrics which resolve the afore-mentioned disadvantages. These are decoded information bit per energy and throughput per area unit. Efficiency metrics are assessed by various implementations of turbo decoders, LDPC decoders and convolutional decoders. New exploration methodologies are presented, which permit an appropriate benchmarking of implementation efficiency, communications performance, and flexibility trade-offs. These exploration methodologies are based on efficiency trajectories rather than a single snapshot metric as done in state-of-the-art approaches.

Index Terms

Channel coding, algorithmic complexity, energy efficiency, design space exploration, design methodology.

I. Motivation

Today, high-end smart phones have to support multiple radio standards, advanced graphic- and media applications and many other applications resulting in a workload of about 100 giga operations per second in a power budget of 1 Watt [1]. The baseband processing in the radio part (mainly front-end processing, demodulation and decoding) requires more than 50% of the overall workload in a state-of-the-art 3.5G smart phone. To achieve higher spectral efficiency new transmission techniques like MIMO will be established. However, this will increase the workload even further. Thus there is a strong need for *efficient* wireless baseband receivers. The overall efficiency of a baseband receiver depends on

- *communications efficiency*: expressed by the spectral efficiency and signal-to-noise ratio (SNR). The requirements on the communications efficiency have the largest impact on the selected baseband processing algorithms.

This work has been partly supported by the UMIC Research Center, RWTH Aachen University. F. Kienle and N. Wehn are with the Microelectronic Systems Design Research Group, University of Kaiserslautern, Kaiserslautern, Germany, e-mail: {kienle,wehn} @eit.uni-kl.de H. Meyr is with the Institute for Integrated Signal Processing System, RWTH Aachen University, Aachen, Germany

- *implementation efficiency*: related to silicon area, power and energy. Here, the energy efficiency is the biggest challenge due to the limited available battery power in many devices.
- *flexibility*: in software defined radio, receivers have to support multiple standards and should be configurable at run-time (see software defined radio). There are various silicon implementation styles ranging from general purpose architectures, over DSPs and ASIPs down to fully physically optimized IP blocks which strongly differ in their implementation efficiency but also in their flexibility. For each building block of the receiver a detailed analysis of flexibility requirements has to be carried out to find the best flexibility/cost trade-off. Thus, advanced baseband receivers are heterogeneous multi-core architectures implemented in different design styles.

System requirements are very often specified by communication standards like UMTS, LTE and WiMAX, which define different services in terms of required communications performance and system data throughput, i.e. information bits per second. To obtain an efficient baseband implementation, a careful and elaborate *design space exploration* has to be performed. This is a very challenging task due to the size and the multi-dimensionality of the space. Therefore it is mandatory to prune the design space in an early stage of the design process. In this process the algorithms have to be selected and quantitatively compared to each other with respect to their system performance and implementation efficiencies.

Appropriate metrics are key for efficient design space exploration to measure the algorithmic and the implementation complexity respectively.

A. Algorithmic Complexity

There exists no universal measure for complexity. In computer science the O-notation describes the asymptotic complexity behavior of an algorithm. In information theory the Kolmogoroff complexity is defined by the minimum description length of a string. These measures are inadequate for implementation purposes. A useful description of complexity for our purpose is to use the number of ”algorithmic” operations which have to be performed per received samples by the algorithms of a baseband receiver. This complexity metric has the advantage of being independent of a specific implementation of the algorithms. Based on this complexity definition a two-dimensional graph can be set up in which the horizontal axis correspond to the sample rate of the receiver (which is proportional to the data rate) and the vertical axis corresponds to the operations per sample which have to be carried out. Typically, both axes are scaled logarithmically. An example of such a graph is show exemplarily in Figure 1 for the digital baseband processing of a 384 kbit/s UMTS receiver. The off-diagonal lines describe points of equal number of *operations per second*, often expressed in million of operations per second (*MOPs*)or giga operations per second (*GOPs*).

Several important conclusions can be draw form this figure. First, the receiver task is heterogeneous. There exist a large variety of algorithms ranging from the complex MUSIC algorithm to the simple root raised cosine matched filtering. Note, that the largest number of operations is performed in simple algorithms such as filtering and correlation while the most complex MUSIC algorithm in this example requires less *MOPs*. From this follows

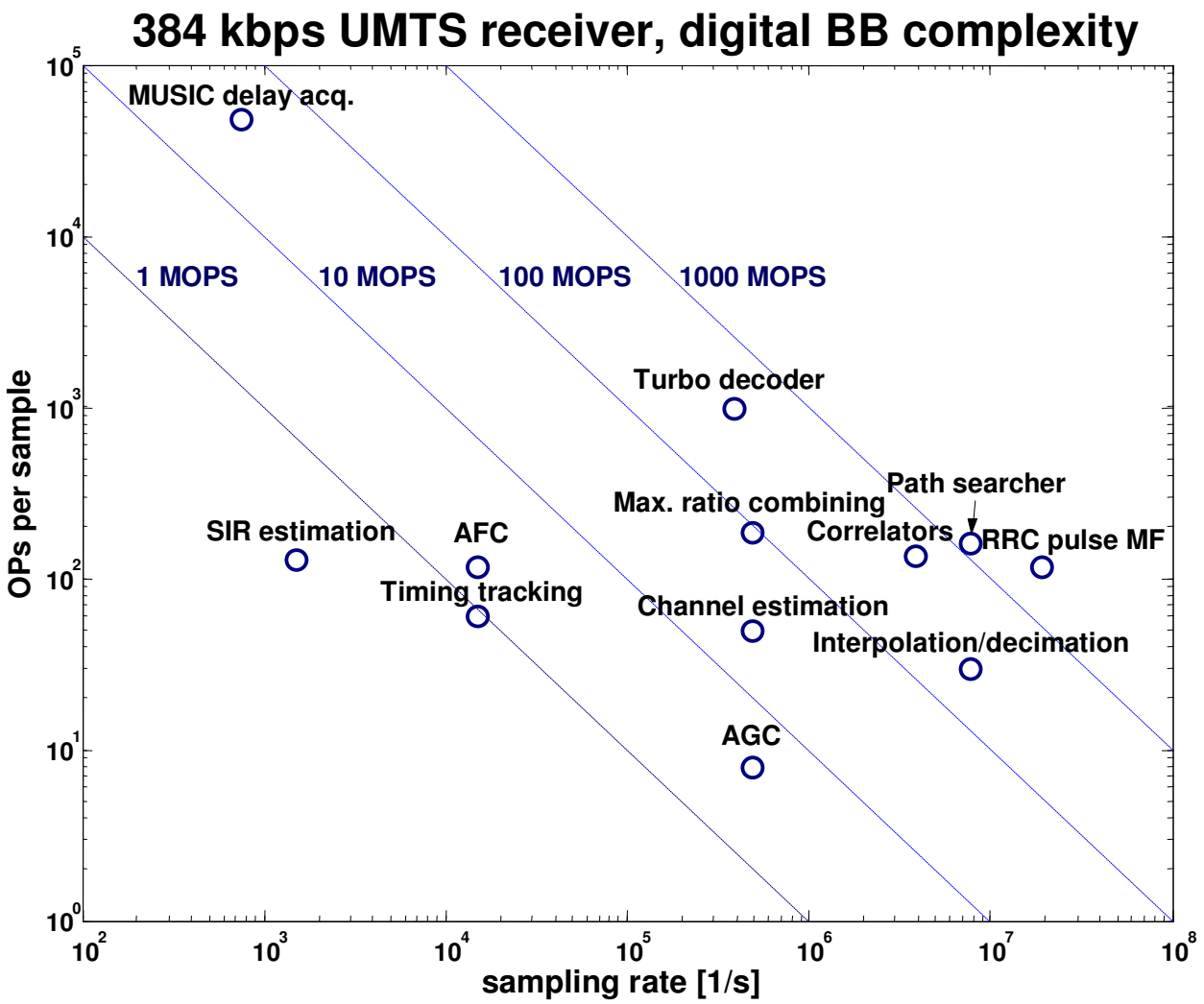


Fig. 1: Operations per sample over sampling rate [1/s]

that counting only operations/sec is entirely misleading. The second conclusion is that the heterogeneity of the algorithms points to architectural features for implementation. Simple algorithms such as filtering and correlation can be implemented very efficiently in architectures requiring little flexibility in the form of parameterizability. Complex algorithms must be programmable and thus require high flexibility. Recently, Van Berkel determined the complexity of various algorithms in baseband processing based on a similar metric. In his remarkable and comprehensive [1] he has shown the number of ”algorithmic” operations which have to be performed per received bit by the algorithms of a baseband receiver for different communication standards. Eberli from ETH Zürich is using a similar metric [2] for measuring complexity in baseband processing by calling these operations ”atomic” operations.

From a communication system point of view we can separate digital processing in the baseband into two parts: the so called ”inner modem” and ”outer modem” [3] respectively. Task of the inner modem is the extraction of

symbols from the received signal waveform, i.e., equalization, channel estimation, interference cancellation and synchronization. The outer modem performs demodulation, de-interleaving and channel decoding on the received symbols. Thus algorithmic complexity in baseband processing is normally separately plotted for the inner and the outer modem respectively. A large diversity exists in the various baseband processing algorithms with respect to operation types, operation complexity, and data types especially between the inner and out modem. Figure 2 in [1] shows the algorithmic complexity for the inner and outer modem measured in giga operations per second (GOPs). It can be seen that sophisticated decoding schemes like turbo and LDPC codes utilized in advanced services like LTE require much more operations than the algorithms of the inner modem.

B. Implementation Complexity

On the implementation side a strong emphasis has to be put on the energy efficiency. Implementation complexity and algorithmic complexity are strongly interrelated in wireless baseband processing. Thus they have to be related to each other. Often it is argued that the implementation complexity is directly related to the algorithmic complexity. E.g. Eberli [2] considers the implementation complexity by introducing a cost factor for each atomic operation which reflects its implementation cost. For design space exploration, graph representations are commonly used:

- A two dimensional energy efficiency graph: one axis corresponds to the algorithmic complexity, e.g. measured in $GOPs$, and the other axis to the power, e.g. measured in mW, consumed when providing the corresponding operations/second. Each point in this graph describes the *energy efficiency metric*, i.e. *operations/second/power unit*, usually measured in $GOPs/mW$. Since energy corresponds to power multiplied with execution time, each point gives the *operations/energy* measured in $operations/Joule$.
- In a similar way we can set up an area efficiency graph in which one axis represents the needed area. Each point in this graph yields the *area efficiency metric*, i.e. *operations/second/area unit*, usually measured in $GOPs/mm^2$.

Note that the energy and area efficiency for the same algorithmic complexity can vary by several orders of magnitude, dependent on the selected implementation style. By far the highest energy efficiency is achieved by physically optimized circuits, however, at the expense of no flexibility. The highest flexibility via software programmability at the expense of low energy efficiency is achieved by digital signal processors. The designer has to find a compromise between the two conflicting goals by trading off flexibility vs. energy efficiency. Flexibility is hard to quantify. The optimum design point is thus to be understood qualitatively. It depends on the application and a large number of economic and technical considerations. We can combine energy and area efficiency in a two dimensional design space in which the two axis correspond to area and energy efficiency respectively. This is a well known representation of the design space.

C. Assessment of Metrics

Area, throughput and especially energy in many system-on-chip implementations are dominated by data-transfers and storage schemes [4] and not by the computations itself. However common metrics as described above are

focusing solely on operations, and are not considering data-transfer and storage issues at all. Thus, these metrics are only valid if the operations dominate the implementation complexity. This is the case in data-flow dominated algorithms like an FFT calculation, correlation or filtering.

Most algorithms in the inner modem of baseband receivers belong to this class of algorithm. However the *channel decoding algorithms* in the outer modem largely differ from the algorithms used in the inner modem. Here, the operations to be performed are non-standard operations (e.g. tanh) using non-standard data types (e.g. 7 bits). But more important, the overall implementation complexity, especially energy, is dominated by data-transfers and storage schemes. A change in the algorithm with respect to computation, e.g. optimal versus suboptimal algorithm by approximating computations, has only a minor impact on the energy efficiency as shown later.

The transitions from 3G to LTE advanced require 2 orders of magnitude improvement in energy efficiency. This improvement will come to a small extent from technology scaling [5]. Efficient system-on-chip implementations are feasible when channel coding schemes and the corresponding decoding algorithms are co-designed together with the architecture (architecture aware code design) [6] [7] [8]. This is in accordance with a general trend towards co-design algorithm and architecture in receiver design realizing that the traditional separation of algorithm and architecture design leads to suboptimal results. In channel decoding the co-design focuses on data-transfer and storage schemes. Examples are special interleavers for turbo codes (e.g. LTE standard [9]) and special structures of the parity check matrix for LDPC codes (e.g. DVB-S2 standard [10]). These special structures allow an efficient parallel implementation of the decoding algorithm with small overhead in data-transfer and storage. GOPs based metrics do not at all reflect such specific structures.

Another important issue is flexibility. Flexibility on the algorithmic side, e.g., code rates and block sizes in the case of channel decoding, have a large impact on the implementation complexity. By looking only on the operations in the algorithm, flexibility is normally not considered.

In summary, efficiency metrics based on GOPs are questionable. Particularly, for non-data flow dominated algorithms since they entirely neglect important issues like data and storage complexity, algorithm/architecture co-design and flexibility.

In this paper we focus on channel decoding as application. The contributions of this paper are:

- we will show that the GOPs metric yields wrong conclusions.
- we will introduce suitable metrics for energy and area efficiency.
- we present a methodology for design space exploration based on these metrics.

II. Reference Designs

Reference designs are key to assess various metrics. Thus, we selected 5 different channel decoder implementations which our research group has designed in the last couple of years. Using own design has the advantage that all data are available. The decoders differ in services (throughput, block sizes, code rates), decoding algorithms, flexibility and implementation styles. Selected codes are convolutional codes, turbo codes and LDPC codes. The 5 different decoders are:

Decoder	**Flexibility**	**Max. Block-size**	**Throughput [Mbit/s]**	**Frequency [MHz]**	**Area [mm2]**	**Dynamic Power [mWatt]**
ASIP [11]	Conv. Codes Binary TC Duo-binary TC	N=16k	40 14 (6iter) 28 (6iter)	385 (P&R)	0.7 (P&R)	˜100
LTE turbo [12]	R=1/3 to R=9/10 by puncturing	N=18k	150 (6.5 iter)	300 (P&R)	2.1 (P&R)	˜300
LDPC flexible	R=1/4 to R=9/10	N=16k	30 (R=1/3 40iter) 100 (R=1/2 20iter) 300 (R=0.83 10iter)	385 (P&R)	1.172 (P&R)	˜389
LDPC WiMedia 1.5 [13]	R=1/2-4/5	N=1.3k	640 (R=1/2 5iter) 960 (R=0.75 5iter)	265	0.51	˜193
CC Decoder	64-state NSC		500	500	0.1	˜37

TABLE I: Reference decoders: service parameters and implementation results in 65nm technology

- An application specific instruction set processor (ASIP) [11] capable of processing binary turbo codes, duo-binary turbo codes and various convolutional codes with different throughputs dependent on code rate and decoding scheme.
- A turbo decoder which is LTE [9] compliant. The maximum throughput is 150Mbit/s at 6.5 decoding iterations.
- An LDPC decoder optimized for flexibility, supporting two different decoding algorithms, code rates from R=1/4-9/10 and a maximum block length of 16384.
- An LDPC decoder which is WiMedia 1.5 compliant and optimized for throughput, supporting code rates from R=1/2-4/5 with two block lengths N=1200 and N=1320 bits [13].
- A convolutional decoder with 64-state which is WiFi [14] compliant.

All decoders are synthesized on a 65nm CMOS technology under worst case conditions with $V_{dd} = 1.0V, 120C$. Power estimations are based on nominal case $V_{dd} = 1.1V$. Table I gives an overview of the key parameters of the different decoders. P&R indicates that the corresponding data are post-layout data. The payload (information bits) throughput depends on the number of decoding iterations for turbo and LDPC codes which also impacts the communications performance. Thus, the throughput is specified dependent on the number of iterations.

In Table II we show the number of algorithmic operations required to process the different types of convolutional codes, turbo codes and LDPC codes. Bit-true C-reference models are used for operation counting. All operations were normalized to an 8 bit addition. The number of operations is related to one information bit which has to be decoded, i.e., operations/information bit. The total number of operations which have to be performed per second, i.e. GOPs, depends on the code rate R and throughput which depends on the number of iterations for LDPC and turbo codes. Two different algorithms for LDPC codes were investigated. Both algorithms are suboptimal algorithm approximating the belief propagation algorithm: the Min-Sum algorithm with a scaling factor and the

Code	**operations/bit**		**GOPs (w.r.t. throughput)**		
	#iter	#(op/bit)	100 Mbit/s	300 Mbit/s	1 Gbit/s
CC (states=64)		200	20	60	200
LDPC Min-Sum	5 iter	75/R	7.5/R	22.5/R	75/R
	10 iter	150/R	15/R	45/R	150/R
	20 iter	300/R	30/R	90/R	300/R
	40 iter	600/R	60/R	180/R	600/R
Turbo (Max-Log)	2 iter	280	28	84	280
	4 iter	560	56	168	560
	6 iter	840	84	252	840

TABLE II: Number of normalized algorithmic operations per decoded information bit for different channel decoders dependent on throughput and code rate R.

λ-3-Min algorithm [15] which is a more accurate approximation. However the latter one needs about 3.3 times more operations. This more accurate approximation is mandatory if lower code rates $R < 0.5$ have to be supported like in DVB-S2 decoders [16]. In Table II we have only listed the operations for the Min-Sum algorithm. To obtain the operations for the λ-3-Min algorithm, operations and GOPs have to be multiplied by 3.3 respectively. The flexible LDPC decoder in our reference design was designed for both decoding algorithms, the WiMedia LDPC decoder is based on the Min-Sum algorithm only.

III. Suitable Metrics

The energy efficiency graph for our reference designs is shown in Figure 2. It can be seen that the energy efficiency, measured in GOPs/mW, largely varies for the different decoders.

The two dimensional design space, covering area and energy efficiency, is illustrated in Figure 3. In this graph, efficient architectures w.r.t. area and energy have to be located in the upper right corner. Less efficient architectures are placed in the lower left corner.

We see that the convolutional decoder appears to be the most efficient decoder while the ASIP being the decoder with the lowest efficiency. One interesting observation is the efficiency of the flexible LDPC decoder. The efficiency largely increases in both directions (area, energy) when replacing the Min-Sum by the λ-3-Min algorithms which is the more complex algorithm in terms of operations. As described in the previous section the GOPs for this algorithm increases by a factor 3. We would expect a large increase in area and power accordingly. However, the area and power only increases by about 10% for the λ-3-Min algorithm. This is due to the fact that area and energy in both decoders are dominated by the data-transfer and storage scheme and the change in the operations of algorithm has only a small impact on it. In other words, the number of operations increases much larger than the implementation complexity. This is a hint that a GOPs based metric is not suited. Moreover we see that the λ-3-Min based flexible

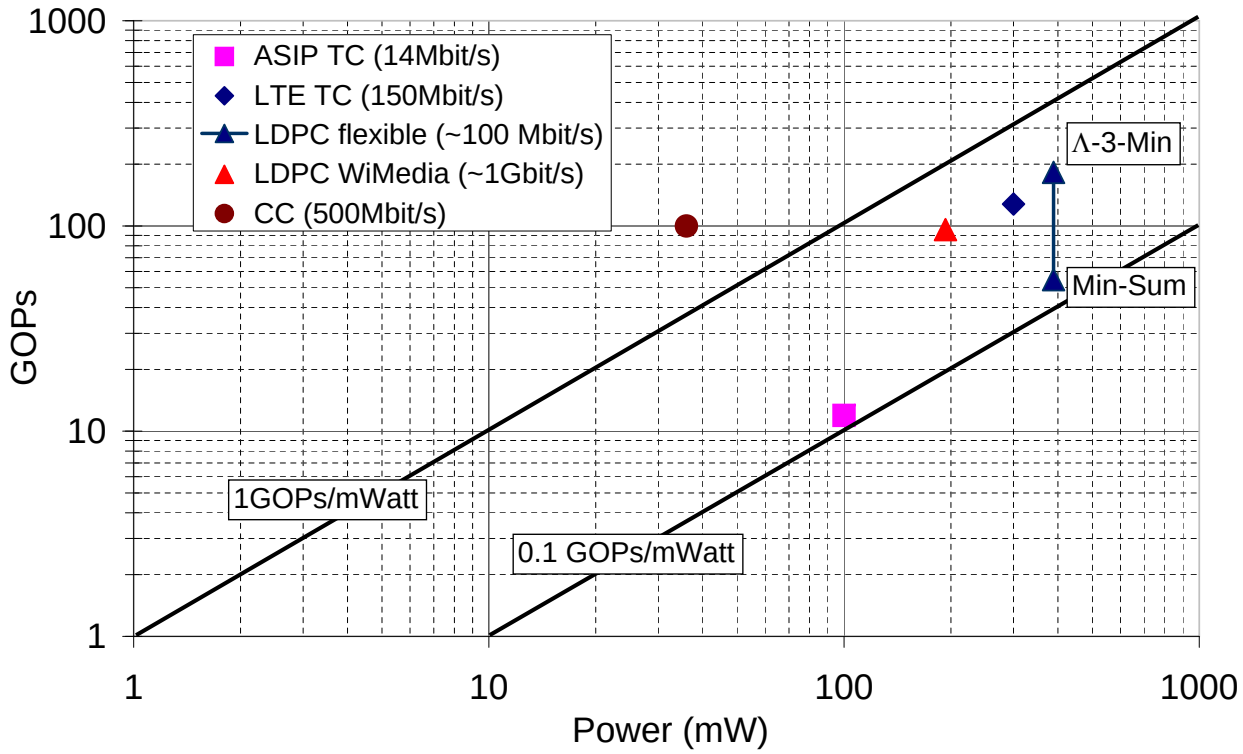


Fig. 2: Operations/second versus power

decoder has nearly the same efficiency as the less flexible WiMedia decoder which is optimized for throughput. We would expect that such a less flexible for throughput optimized decoder has a higher efficiency compared to the flexible one.

In the following we introduce metrics to resolve the afore mentioned anomalies. Instead of using the operations which have to be carried out for processing per task we normalize to the number of information bits per task. Metrics normalized to the number of information bits have the following properties. They allow comparing

- competing architectures for a given algorithm since the efficiency metrics are independent of the specific operations and data types used to execute the algorithm. All implementation issues like data-transfer and storage are taken into account since the metrics is oblivious to how the task has been executed.
- different coding schemes as a function of the communication parameters (modulation, signal to noise ratio, bandwidth).

In particular, iterative decoding algorithm can be compared in a meaningful way to non-iterative algorithm. Energy efficiency is a multidimensional problem. The performance of the physical layer of a communication system depends on the transmit energy via the SNR at the receiver, denoted *communication energy*, and the processing energy in the receiver to retrieve the information, denoted *computation energy*. There exists an interesting trade off between communication and computation energy which has to be exploited in advanced, adaptive systems. For example,

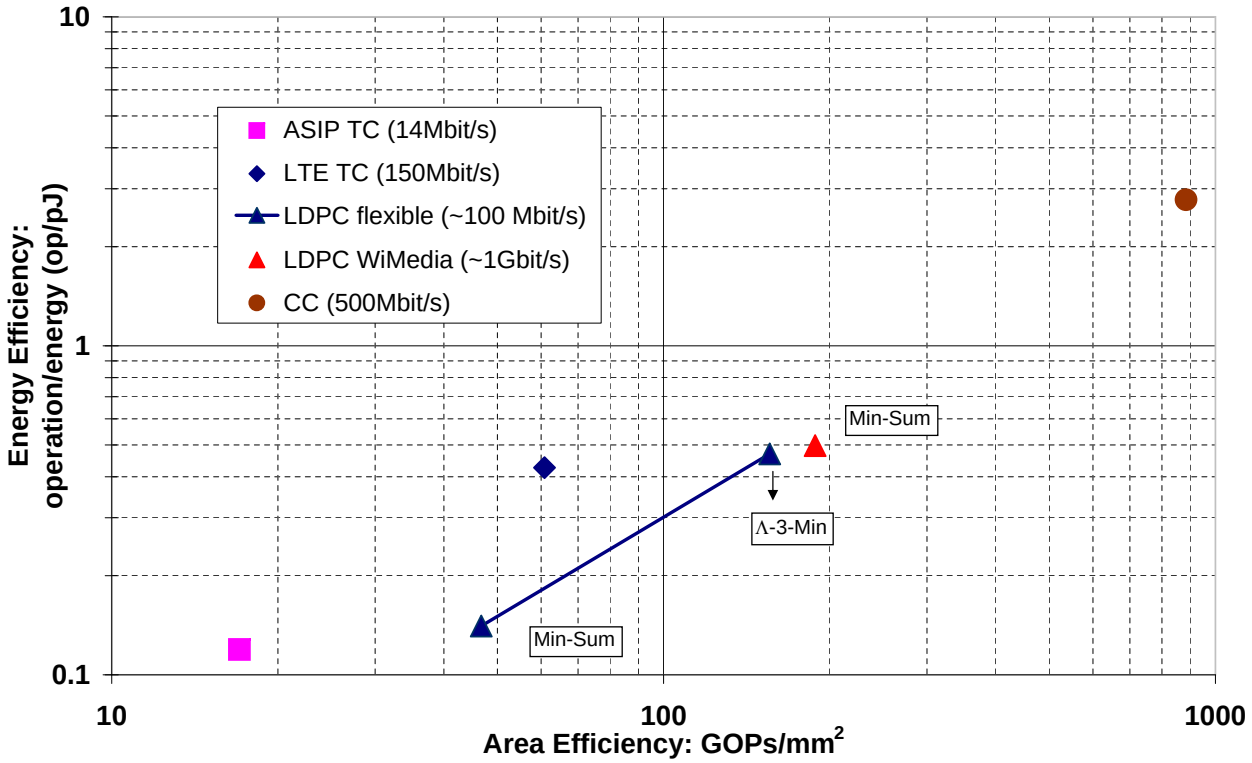


Fig. 3: Energy efficiency versus area efficiency based on operations

iterative algorithm operates at much lower SNR than convolutional codes. Decreasing the SNR, however, results in an exponentially increasing complexity and energy consumption due to the large number of iterations required for decoding.

We define the two suitable metrics for implementation efficiency as follows

- energy efficiency metric: *decoded information bit per energy* measured in bit/nJ
- area efficiency metric: *information bit throughput per area unit* measured in $Mbit/s/mm^2$

We have mapped our decoders to the design space which is based on these metrics, see Figure 4. Again efficient architectures are placed in the upper right corners, inefficient architectures in the lower left corner.

A large change in the relative and absolute positions can be observed for some decoders, when comparing them with figure Figure 3.

- The difference in the efficiency between the two instances of the flexible LDPC decoder (Min-Sum and the λ-3-Min decoder respectively) is now much smaller. Moreover the Min-Sum decoder is more efficient than the other ones which was not the case in the conventional design space. This matches our expectations since the data-transfer and storage scheme in both decoders is nearly identical and the increase in computation results in only a small energy and area increase as described above. Both decoders are targeting the same throughput.
- The efficiency of the WiMedia decoder which is optimized for throughput and less flexibility, is now much

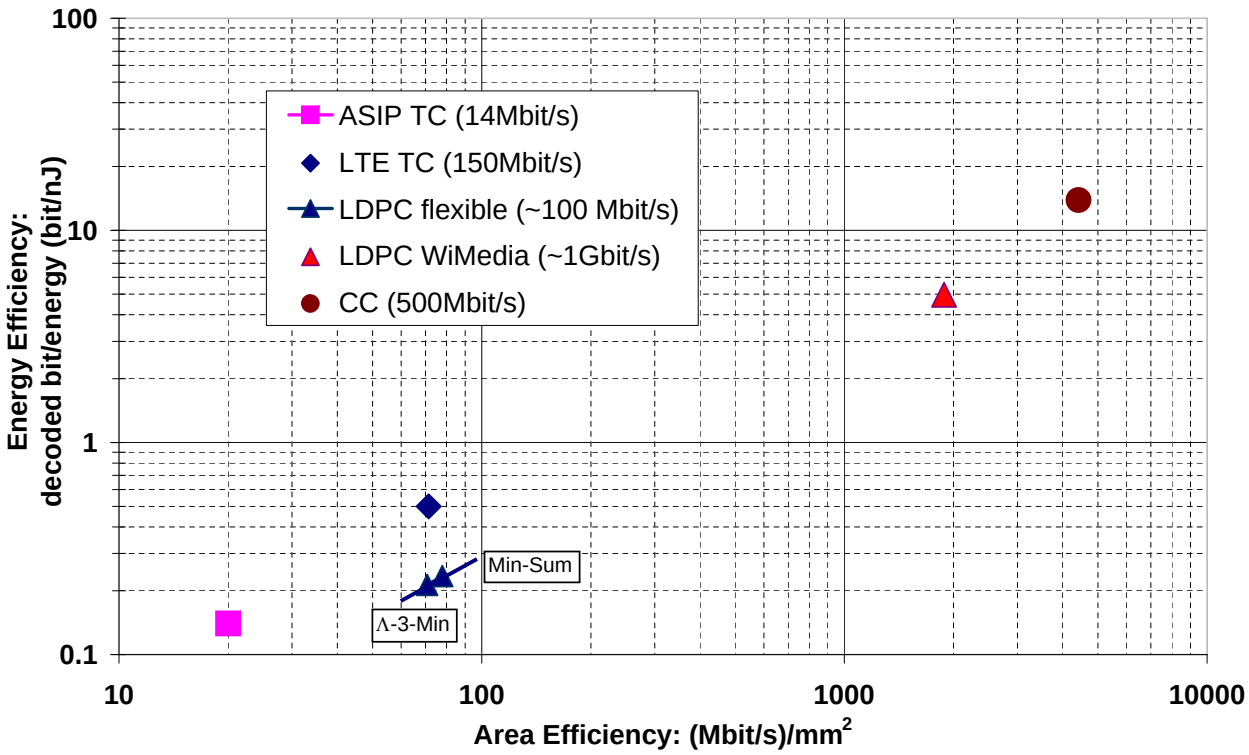


Fig. 4: Design space based on suitable metrics. Decoded information bit per energy over information bit throughput per area unit.

larger than the efficiency of the flexible LDPC decoder which again matches what we expected.

So far we have focused on the implementation complexity but have not discussed the important aspect of flexibility and communication performance. In the following we will investigate the relationship between communication performance, flexibility and implementation efficiency.

IV. Methodology

In the previous section we investigated the absolute and the relative positions of the different decoders to each other. However equally important in this space is the trajectory when specific parameters are changed since such trajectory represents the impact of a specific parameter on the decoder efficiency. The following parameters will be considered: the frame error rates (FER), i.e. communications performance, coding techniques, code rates, number of iterations and throughput. The resulting trajectories well illustrates the strong dependency between communications performance and implementation efficiency. We present two design space exploration methodologies. The first one is driven by implementation efficiency and compares non-iterative decoding techniques (convolutional codes) with iterative decoding techniques (LDPC codes). The second exploration compares two different iterative decoding techniques (LDPC and Turbo codes) with code rate flexibility.

A. Implementation driven design space exploration

We use the current WiMedia 1.5 standard for demonstration. WiMedia features low complexity devices for UWB communication. Thus the WiMedia 1.2 standard used convolutional codes as channel coding technique. In the definition phase of the next generation standard, WiMedia 1.5, LDPC codes were considered as a promising candidate due to their much better communication performance. A throughput of 960 Mbit/s at code rate $R = 0.75$ was defined in the standard. A code/architecture co-design approach [13] resulted in an LDPC decoder which has a much higher efficiency than the flexible LDPC decoder. Note that its efficiency is lower in both dimensions compared to a convolutional decoder. However this comparison completely neglects the communications performance. Five decoding iterations can be maximally performed by the LDPC decoder to comply with the throughput fixed in the standard. As already pointed out, the number of iterations strongly impacts the performance of the LDPC decoder. The frame error rate as a function of the number of iteration is contrasted with implementation efficiency in Figure 5. Point 3 in the design space figure corresponds to the WiMedia 1.5 decoder when performing 5 iterations (this was the decoder assumption in the previous figures when we referred to the WiMedia LDPC decoder). The communication figure shows that this decoder has a 4dB better communication performance than the convolutional decoder. The communication performance is comparable to that of the convolutional decoder if the LDPC performs only two iterations instead of five (case 2 in Figure 5). Finally executing only one iteration in the LDPC decoder results in a communication performance which is about 4dB worse than the convolutional decoder (case 1 in Figure 5).

Important is the resulting trajectory in the design space for the different cases. Two cases have to be distinguished:

- The system throughput is not changed w.r.t. WiMedia 1.5. constraint (scenario a in Figure 5). In this scenario only the energy efficiency is improved (points $3 \rightarrow 2a \rightarrow 1a$). Obviously the decoding time decreases when the decoder executes a smaller number of iterations resulting in a negative time lag. This time lag can be exploited for energy efficiency improvement. For example clock and the power supply could be completely switched off when decoding is finished. This reduces energy and leakage current. Another possibility is to slow down the frequency (frequency scaling). This reduces the energy by the same amount as in the previous case but the peak power consumption during decoding instead of leakage is minimized. The most efficient technique is voltage scaling in which the voltage is reduced which results in the highest energy efficiency.
- The system throughput is changed (scenario b in Figure 5). In this scenario the area efficiency increases by the same amount as the throughput increases due to smaller number of iterations (points $3 \rightarrow 2b \rightarrow 1b$).

We see that the efficiency of the LDPC decoder is increasing with decreasing communication requirements, i.e. number of iterations. Thus, the *decoder efficiency is represented by a trajectory instead of a single point* in the design space. This trajectory results form varying communication performance requirements. We also see that the efficiency of the LDPC decoder outperforms the convolutional decoder at the same communication performance.

B. Communications performance driven exploration

In the previous exploration we have compared implementations efficiency between non-iterative and iterative decoding techniques dependent on throughput and frame error rate behavior for *fixed code rates*. In this section we

compare two iterative decoding techniques and put emphasis on code rate flexibility and dependencies. Reference is an LTE turbo decoder implementation. This LTE turbo decoder is compared with a flexible LDPC decoder which supports code rate flexibility. The right graph in Figure 6 shows the communication performance for the two decoding schemes dependent on code rates ($R = 0.5$ and $R = 0.83$) and iteration numbers. The number of information bits is $K = 6140$ in all cases. Frame error rates are based on fixed point simulations matching the hardware implementation.

We use the communications performance of the turbo decoder with 6.5 iterations as reference point for both code rates. The 6.5 iterations result from the throughput constraint of 150Mbit/s which is specified in the LTE standard. The 6.5 iterations fulfill the LTE communications performance requirements for all code rates.

It is well known that the communication performance in LDPC decoding depends on the number of iterations *and* the code rate. The LDPC decoder under investigations provide large code rate flexibility, i.e., the hardware can support various code rates. The LDPC decoder requires 10 iterations for $R = 0.83$ and 20 iterations for $R = 0.5$ to match the performance of the turbo decoder. For a code rate of $R = 1/3$ even 40 iterations are mandatory (this is not shown in Figure 6b).

Important are the corresponding trajectories in the implementation space. The turbo decoder efficiency is identical for all code rates (see left graph in Figure 6). Thus we have no trajectory. This is due to the fact that the code rate flexibility is implemented by puncturing which has negligible impact on throughput, area and energy.

However the situations is completely different for the flexible LDPC decoder. For a given communications performance the code rate has strong impact on the number of required iterations. This iteration number influences the implementations efficiency as we have seen in the previous exploration case. But beside this impact via the iteration number, there is also a direct impact of the code rate on the implementation efficiency since lower code rates requires also a more accurate decoding algorithm (λ-3-Min algorithm instead of the less complex Min-Sum algorithm). The resulting trajectory is shown in the left graph of Figure 6. We see that the efficiency increases in both directions with increasing code rate (points $1 \rightarrow 2 \rightarrow 3$).

The important observation in this exploration is the varying implementation efficiency of the flexible LDPC decoder represented by the trajectory. This trajectory results from the required code rate flexibility in the LDPC decoder which is necessary to match the communications performance with respect to a competitive turbo code decoder. We see that analyzing only one code rate, and thus one snap shot, could result in a wrong efficiency conclusions.

The two explorations have shown that implementation efficiency for advanced iterative decoders often results in a trajectory instead of a single point in the design space. These trajectories result from the strong interrelation between communication performance, flexibility and implementation efficiency.

V. Conclusion

Understanding the trade-offs between implementation efficiency, communications performance and flexibility will be key for designing efficient baseband receivers. Meaningful efficiency metrics are mandatory to explore and

evaluate the resulting huge design space. We introduced and discussed suitable energy and area efficiency metrics which are based on decoded information bit per energy and throughput per area unit. Various channel decoder implementations were utilized to examine these efficiency metrics with respect to the achieved communications performance and with respect to the decoder flexibility. The presented methodology allows to systematically compare different realizations by jointly considering: implementation efficiency, communications performance and flexibility.

REFERENCES

[1] C. H. van Berkel, "Multi-core for mobile phones," in *Proc. DATE '09. Design, Automation. Test in Europe Conference. Exhibition*, Apr. 20–24, 2009, pp. 1260–1265.

[2] S. C. Eberli, Ph.D. dissertation, ETH Zurich, Integrated Systems Laboratory, 2009.

[3] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers*. John Wiley & Sons Inc, 1998.

[4] M. Miranda, C. Ghez, E. Brockmeyer, P. Op De Beeck, and F. Catthoor, "Data transfer and storage exploration for real-time implementation of a digital audio broadcast receiver on a Trimedia processor," in *Proc. 15th Symposium on Integrated Circuits and Systems Design*, 9–14 Sept. 2002, pp. 373–378.

[5] C. Rowen, "Energy-Efficient LTE Baseband with Extensible Dataplane Processor Units," in *Proc. 9th International Symposium on Multiprocessor Systems-on-Chips (MPSoC'09)*, Savanna, USA, August 2009.

[6] E. Boutillon, J. Castura, and F. Kschischang, "Decoder-first code design," in *Proc. 2nd International Symposium on Turbo Codes & Related Topics*, Brest, France, Sep. 2000, pp. 459–462.

[7] M. Mansour and N. Shanbhag, "Architecture-Aware Low-Density Parity-Check Codes," in *Proc. 2003 IEEE International Symposium on Circuits and Systems (ISCAS '03)*, Bangkok, Thailand, May 2003.

[8] J. Kwak and K. Lee, "Design of dividable interleaver for parallel decoding in turbo codes," *Electronics Letters*, vol. 38, no. 22, pp. 1362–1364, Oct. 2002.

[9] "3GPP LTE (Long Term Evolution) Homepage." [Online]. Available: http://www.3gpp.org/Highlights/LTE/LTE.htm

[10] European Telecommunications Standards Institude (ETSI), "Digital Video Broadcasting (DVB) Second generation framing structure,channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; TM 2860r1 DVBS2-74r8," www.dvb.org.

[11] T. Vogt and N. Wehn, "A Reconfigurable ASIP for Convolutional and Turbo Decoding in a SDR Environment," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1309–1320, Oct. 2008. [Online]. Available: http://dx.doi.org/10.1109/TVLSI.2008.2002428

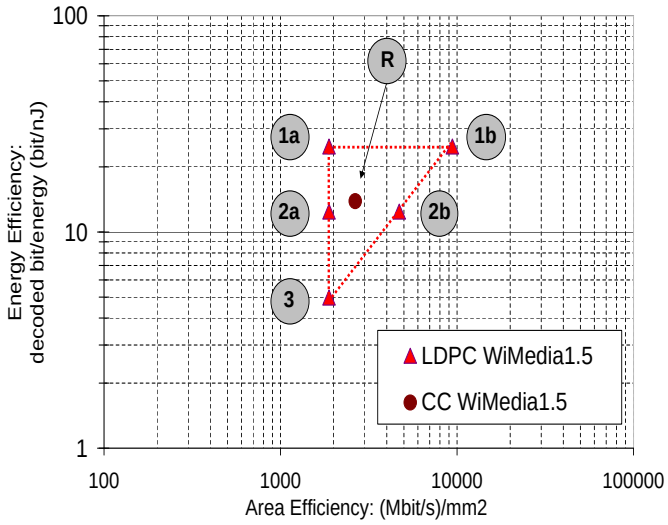
[12] M. May, T. Ilnseher, N. Wehn, and W. Raab, "A 150Mbit/s 3GPP LTE Turbo Code Decoder," in *Proc. Design, Automation and Test in Europe, 2010 (DATE '10)*, 2010, accepted for publication.

[13] M. Alles, F. Berens, and N. Wehn, "A Synthesizable IP Core for WiMedia 1.5 UWB LDPC Code Decoding," in *Proc. IEEE International Conference on Ultra-Wideband ICUWB 2009*, Vancouver, Canada, September 2009, pp. 597–601.

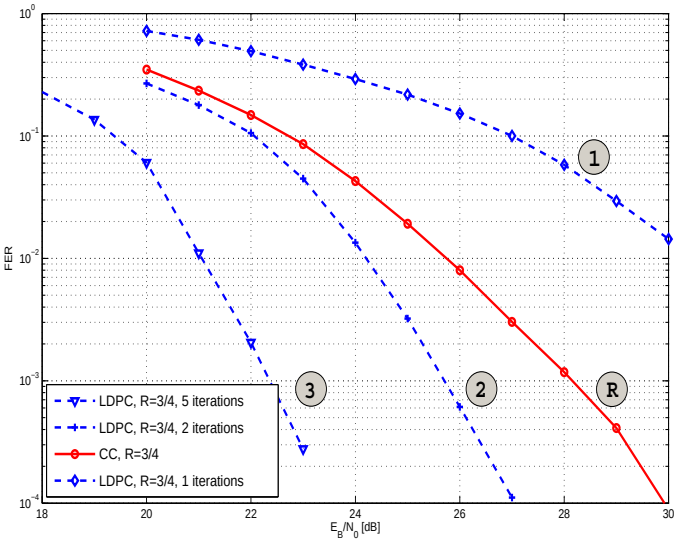
[14] IEEE 802.11, "Wireless Fidelity (Wireless LAN)," http://grouper.ieee.org/groups/802/11/.

[15] F. Guilloud, E. Boutillon, and J. Danger, "λ-Min Decoding Algorithm of Regular and Irregular LDPC Codes," in *Proc. 3nd International Symposium on Turbo Codes & Related Topics*, Brest, France, Sep. 2003, pp. 451–454.

[16] S. Müller, M. Schreger, M. Kabutz, M. Alles, F. Kienle, and N. Wehn, "A novel LDPC decoder for DVB-S2 IP," in *Proc. DATE '09. Design, Automation. Test in Europe Conference. Exhibition*, Apr. 20–24, 2009, pp. 1308–1313.



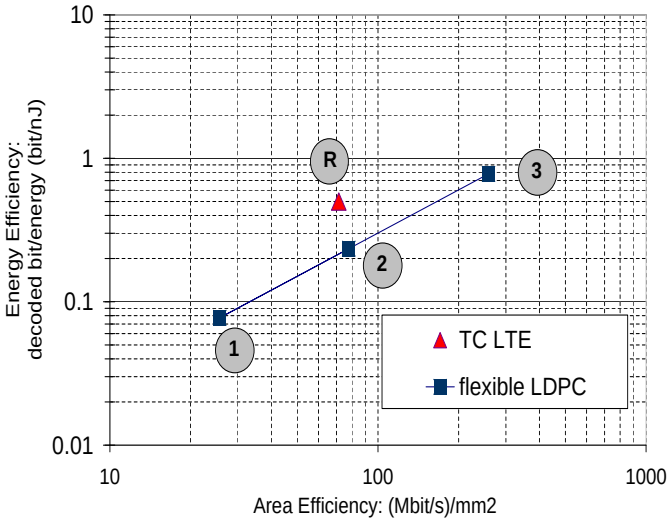
(a) Implementation Efficiency



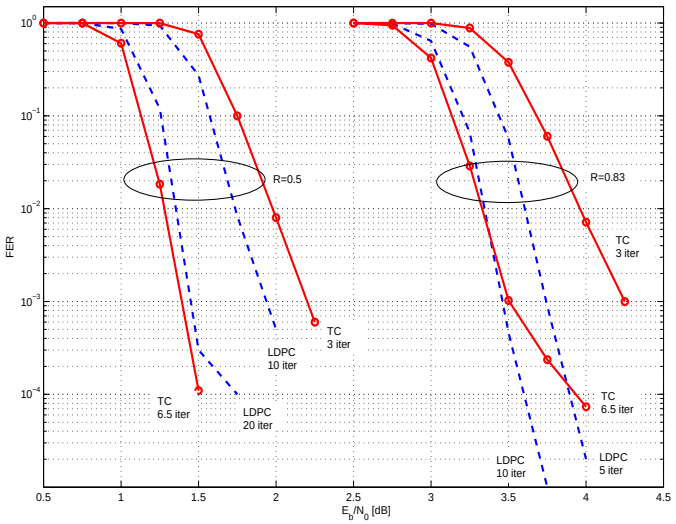
(b) Communications performance (16-QAM, CM1 channel according to IEEE 802.15.3a [13])

R :	Reference convolutional decoder with code rate ($R = 0.75$) and fixed throughput of 1 Gbit/s	
1 :	LDPC code performing 1 iteration	
a)	identical throughput (1 Gbit/s)	$\sim$ 4dB worse communications performance
b)	5 times higher throughput (5 Gbit/s)	$\sim$ 4dB worse communications performance
2 :	LDPC code performing 2 iteration	
a)	identical throughput (1 Gbit/s)	$\sim$ 1dB better communications performance
b)	2.5 times higher throughput (2.5 Gbit/s)	$\sim$ 1dB better communications performance
3 :	LDPC code performing 5 iteration	
	identical throughput (1 Gbit/s)	$\sim$ 4dB better communications performance

Fig. 5: Implementation efficiency and communications performance for WiMedia 1.5 standard



(a) Implementation Efficiency



(b) Communications performance (BPSK, AWGN channel)

R :	Reference of LTE turbo decoder	
	150 Mbit/s throughput for all code rates	reference TC performance at 6.5 iterations
1 :	LDPC code at code rate $R = 1/3$	
	$\sim$ max throughput 30Mbit/s	$\sim$ 40 iterations to match TC performance
2 :	LDPC code at code rate $R = 1/2$	
	$\sim$ max throughput 90Mbit/s	$\sim$ 20 iterations to match TC performance
3 :	LDPC code at code rate $R = 0.83$	
	$\sim$ max throughput 300Mbit/s	$\sim$ 10 iterations to match TC performance

Fig. 6: Implementation efficiency and communications performance of LTE turbo code/decoder and a flexible LDPC decoder.